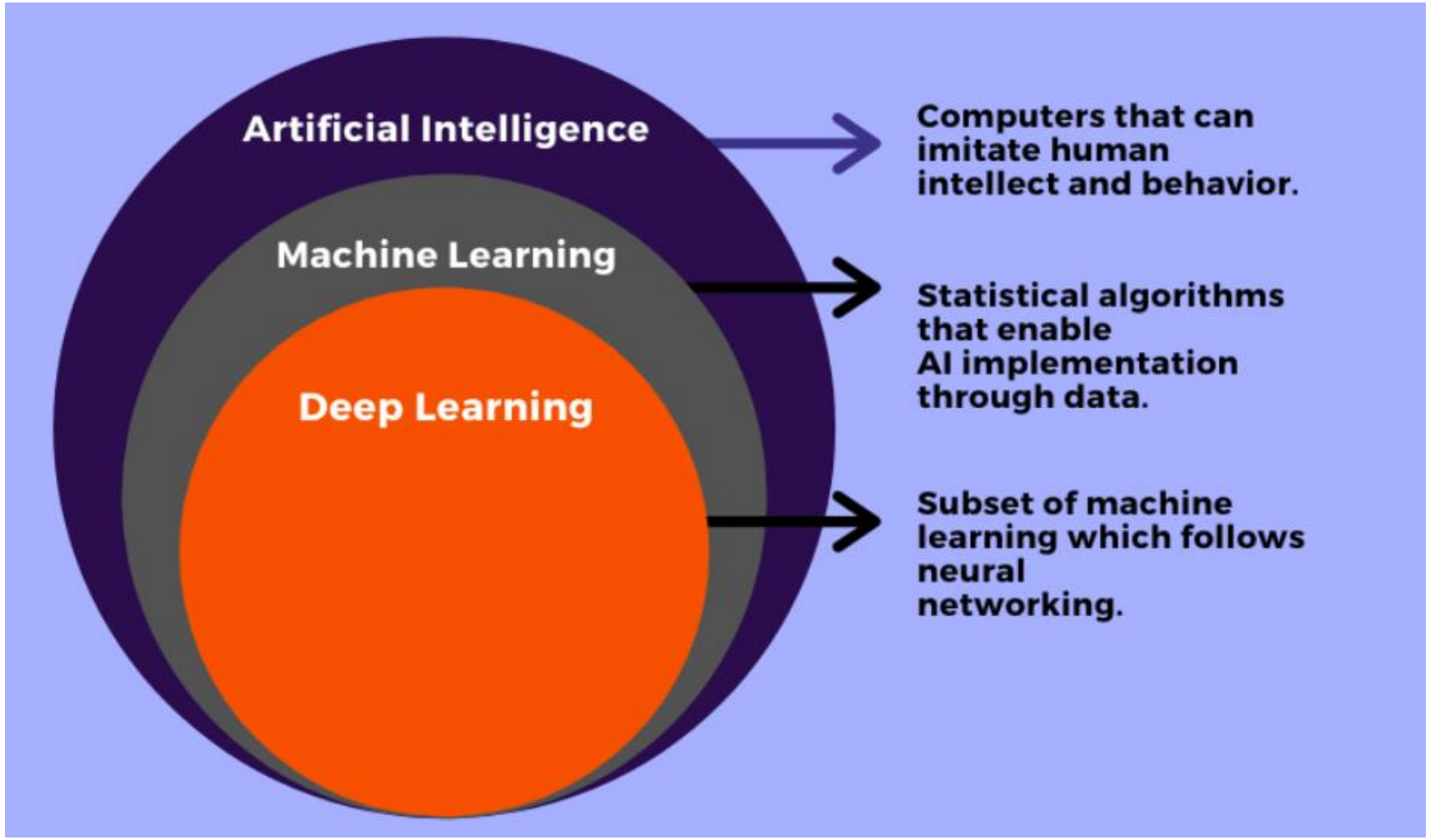
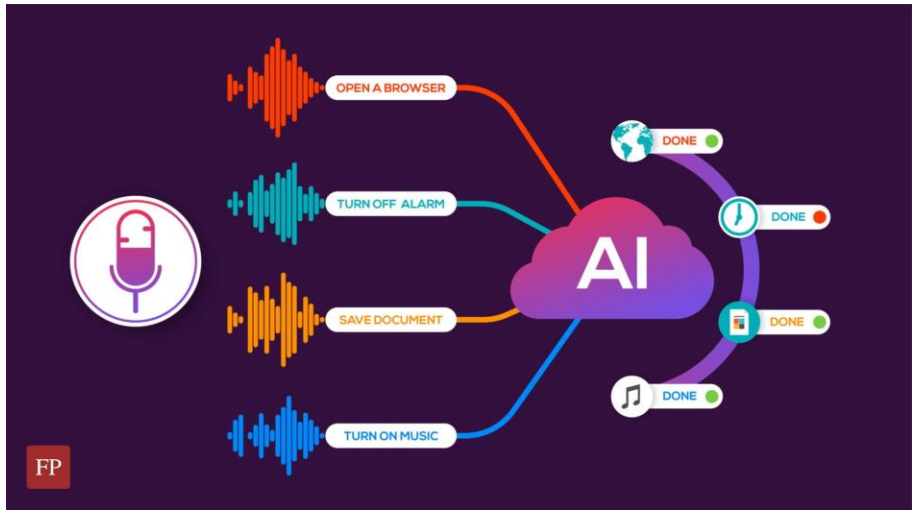
Satellite Image Classification

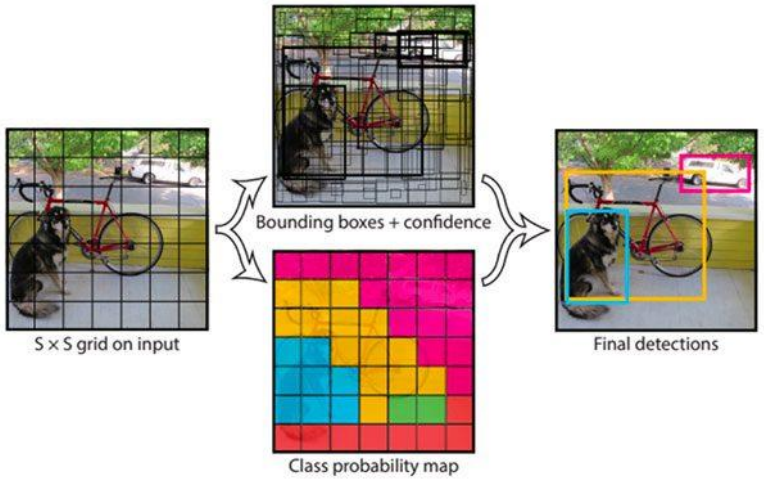
Jann Rovic Cueto[1,2]

[1] Dept. of Creative and Product Design, Asia University, Taiwan;
[2] Dept. of Earth and Space Sciences, Rizal Technological University, Philippines

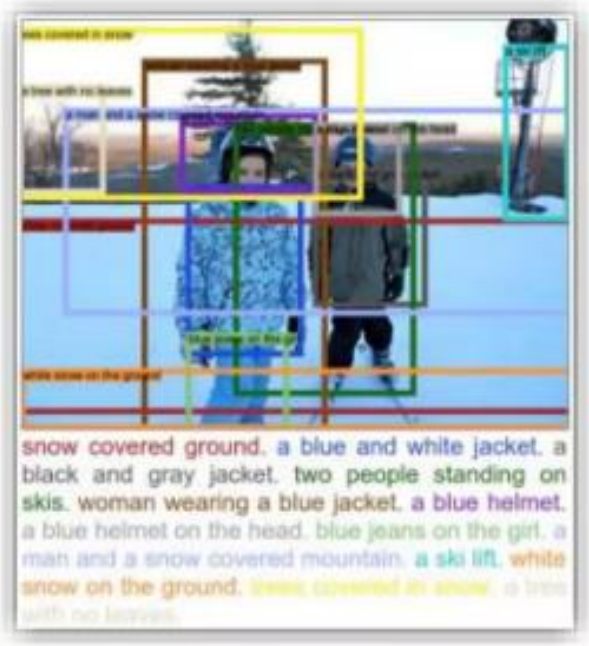
Data Science

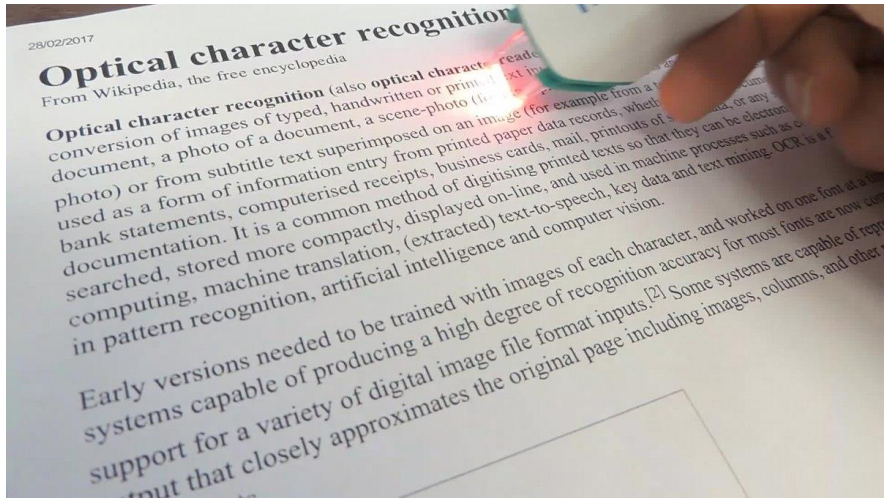






Applications





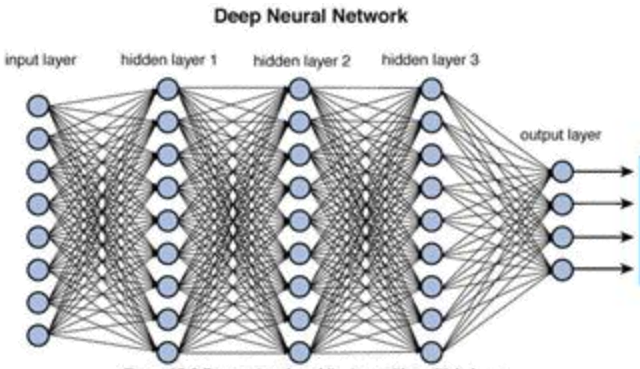


Figure 12.2 Deep network architecture with multiple layers.

Neural Network

Artificial Neural Network (ANN)
Regression and classification

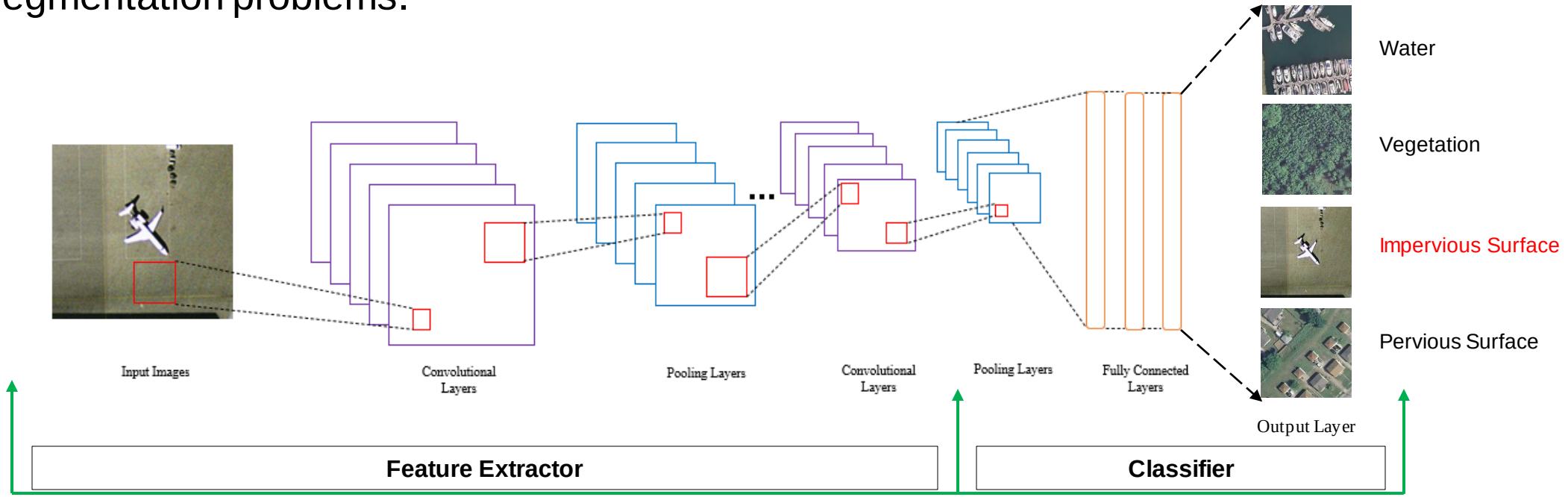
Convolutional Neural Network (CNN)
Computer Vision

Recurrent Neural Network (RNN)
Time series analysis

Convolutional Neural Network

Introduction

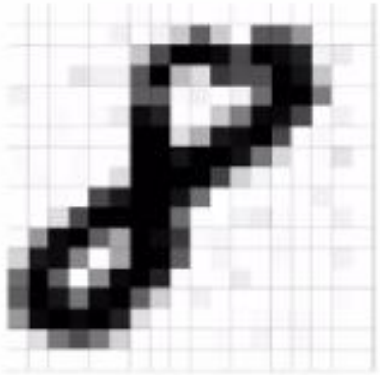
Convolutional Neural Networks (CNNs) learns multi-level features and classifier in a joint fashion and performs much better than traditional approaches for various image classification and segmentation problems.



Convolutional Neural Network

Input

- An Image is a matrix of pixel values.
- If we considered a grayscale image, the value of each pixel in the matrix will range from 0 255.
- If we considered an RGB image each pixel will have the combined values of R,



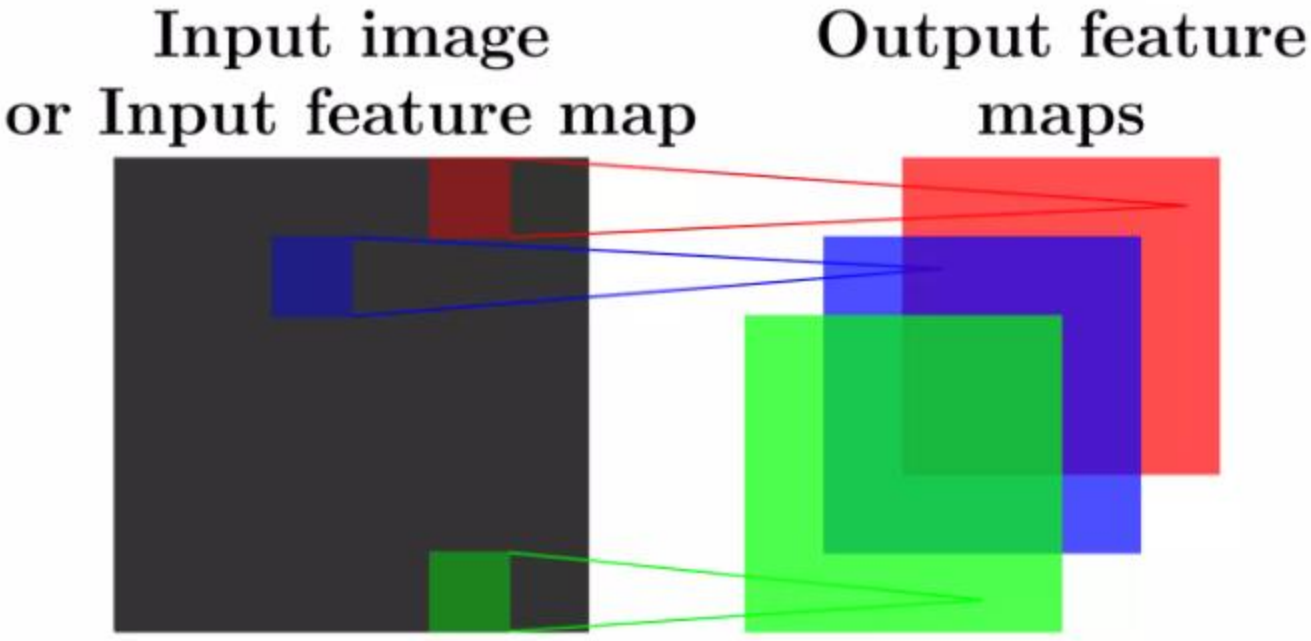
What We See

What Computers See

Convolutional Neural Network

Convolutional Layer

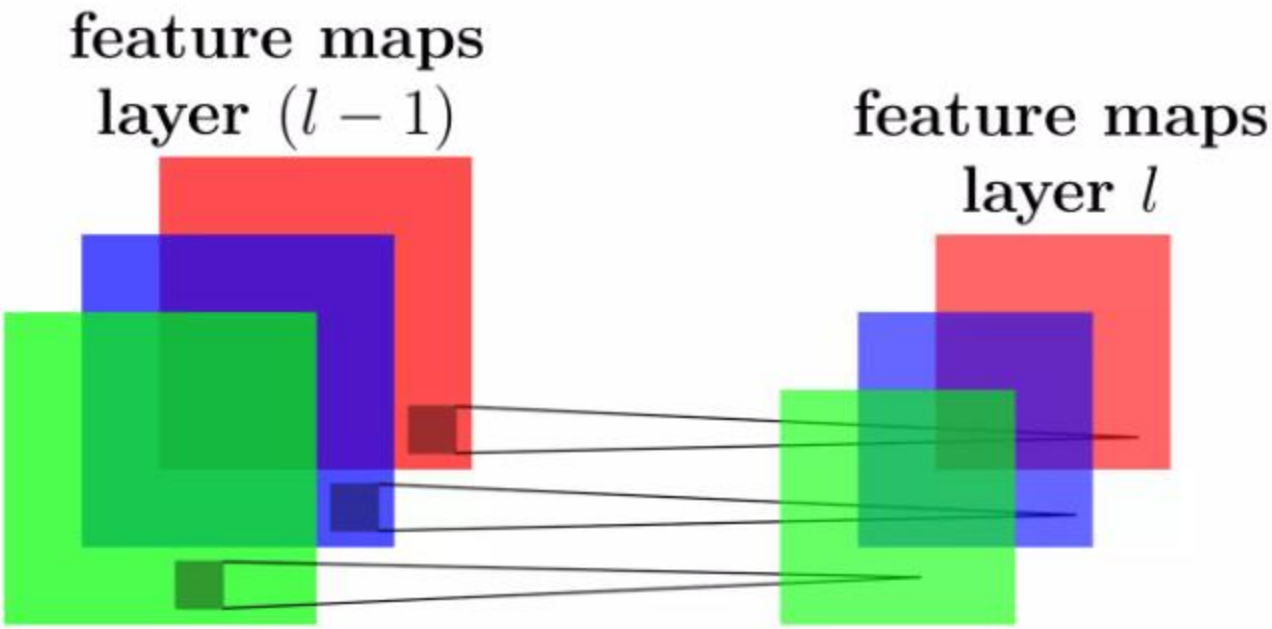
Convolutional Layer acts as a feature extractor that extracts features of the inputs such as edges, corners, and endpoints.



Convolutional Neural Network

Pooling Layer

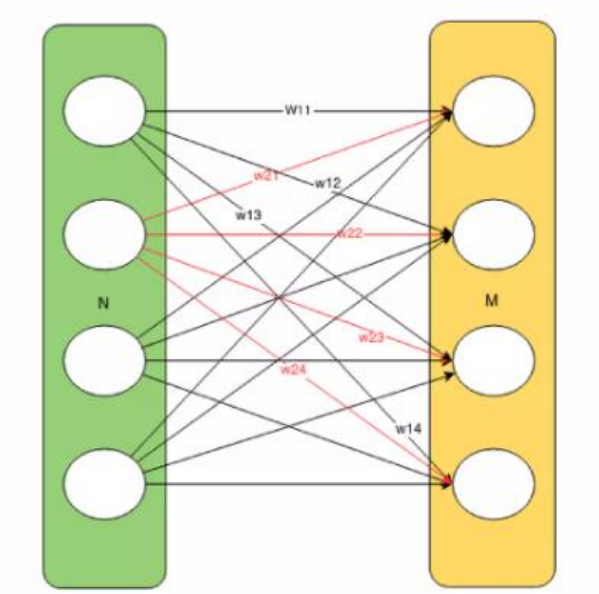
Pooling layer reduces the resolution of the image that reduce the precision of the translation (shift and distortion) effect.



Convolutional Neural Network

Full Connected Layer

- Fully Connected layer have full connections to all activations in the previous layer.
- Fully connect layer act as classifier.



Output Image = (((ImageSize+2*Padding)- KernalSize)/ Stride) +1

Conv 3x3 with stride=1, padding=1

1	2	1	1	1	1
1	1	5	3	9	1
2	4	4	7	5	3
3	6	7	5	6	2
1	6	5	3	1	2
2	3	1	1	1	1

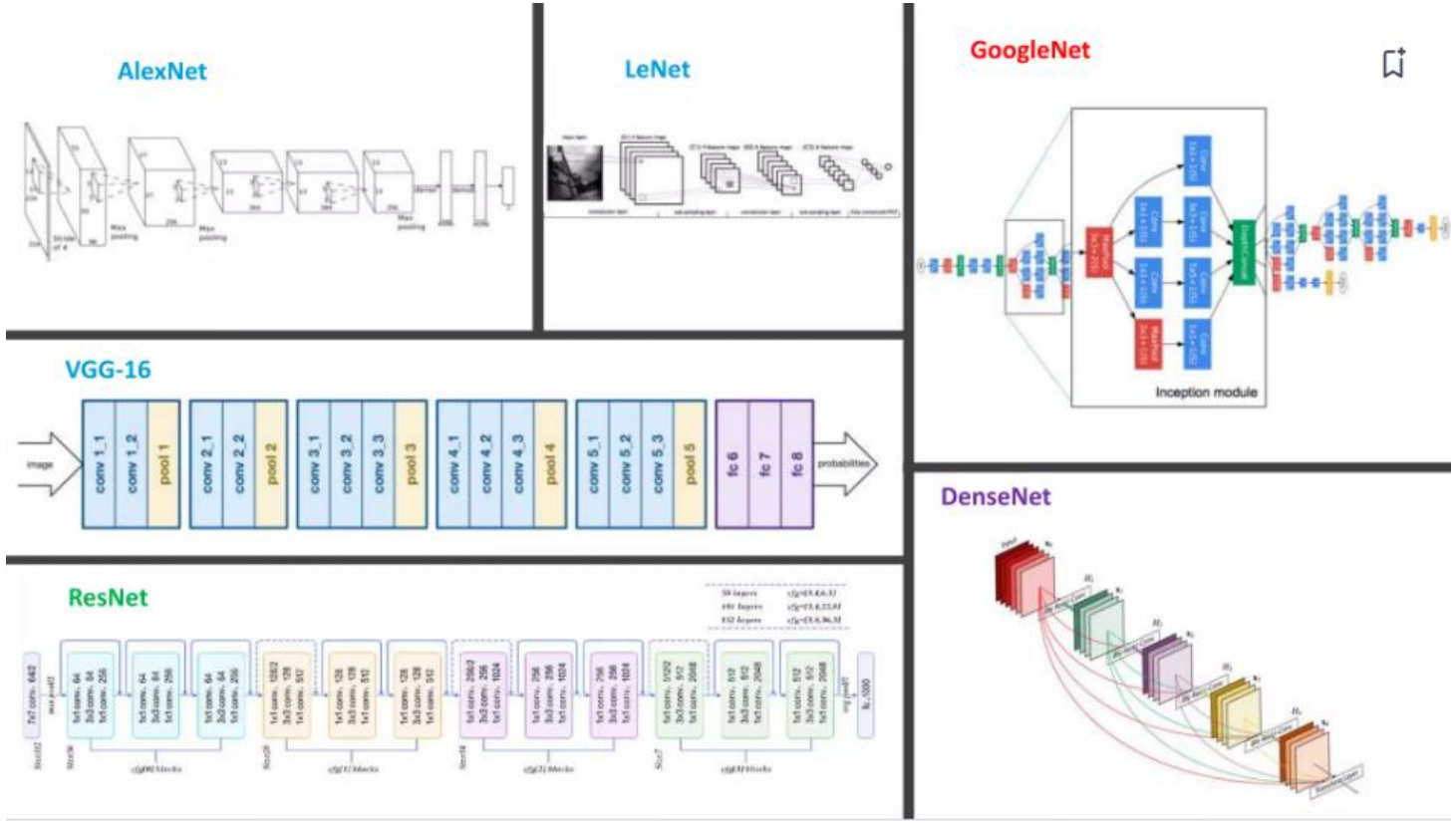
6x6 Image

21	28	36	31
33	42	51	41
38	47	43	34
34	37	30	22

4x4

CNN Architectures

Year	CNN Architecture	Developed By
1998	LeNet	Yann LeCun *et al.*
2012	AlexNet	Alex Krizhevsky, Geoffrey Hinton, and Ilya Sutskever
2013	ZFNet	Matthew Zeiler and Rob Fergus
2014	GoogleNet	Google
2014	VGGNet	Simonyan and Zisserman
2015	ResNet	Kaiming He
2017	DenseNet	Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger



What is K-train Classification?

K-train: a library to help build, train, debug, and deploy neural networks in the deep learning software framework, Keras using TensorFlow.

README Apache-2.0 license

Overview | Tutorials | Examples | Installation | FAQ | API Docs | How to Cite

pypi package 0.41.3 | python 3.7 | 3.8 | 3.9 | 3.10 | 3.11 | License Apache 2.0 | downloads 1M

ktrain

Welcome to ktrain

> a "Swiss Army knife" for machine learning

News and Announcements

- 2024-02-20
  - ktrain 0.41.x is released and removes the `ktrain.text.qa.generative_qa` module. Our OnPrem.LLM package should be used for Generative Question-Answering tasks. See example notebook.

https://github.com/amaiya/ktrain/blob/master/examples/README.md

amaiya/**ktrain**

ktrain is a Python library that makes deep learning and AI more accessible and easier to apply

15 Contributors | 513 Used by | 1k Stars | 268 Forks

https://github.com/amaiya/ktrain

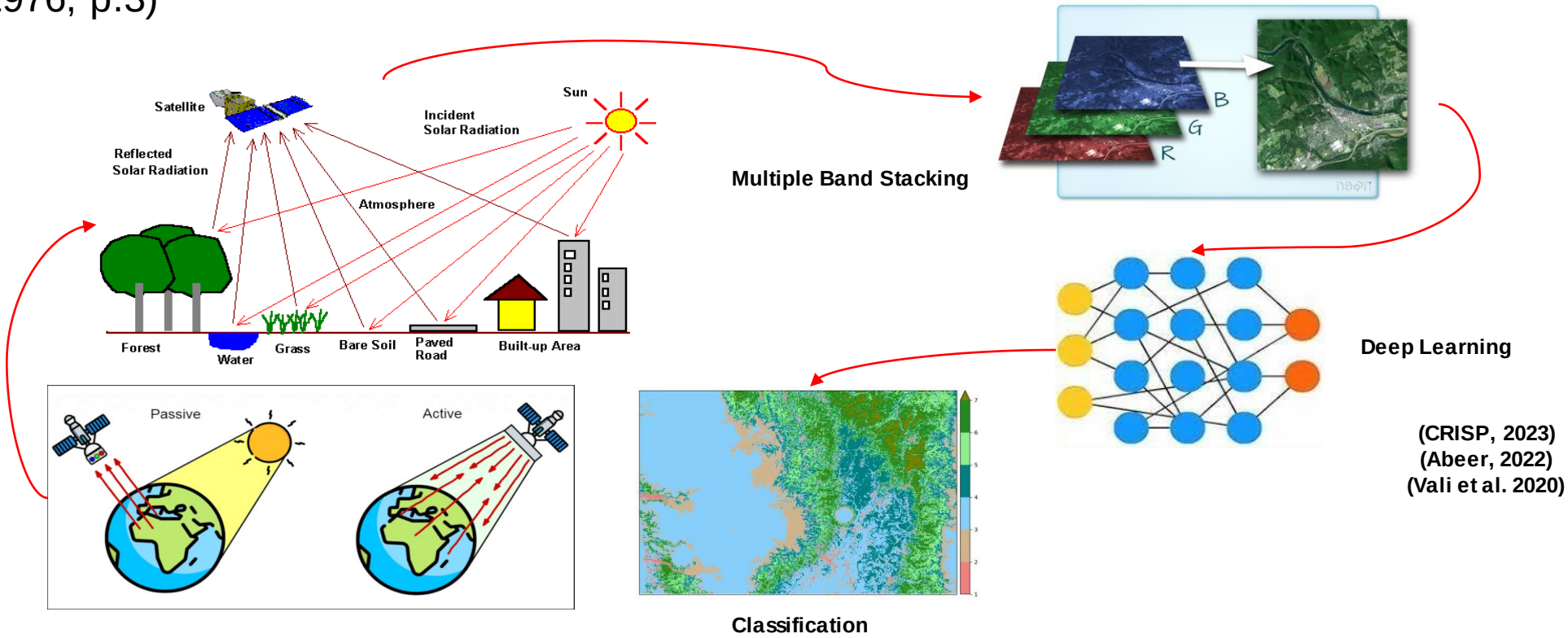
Simply says: *Python library that makes deep learning and AI more accessible and easier to apply to your own dataset.*

Convolutional Neural Networks in Python: Beginner's Guide to Convolutional Neural Networks in Python by Frank Millstein (Ebook) - Read free for 30 days. (2019). In Everand.

What is Remote Sensing?

Remote sensing is the acquisition of physical data of an object without touch or contact.(Lintz and Simoneet, 1776, p.1)

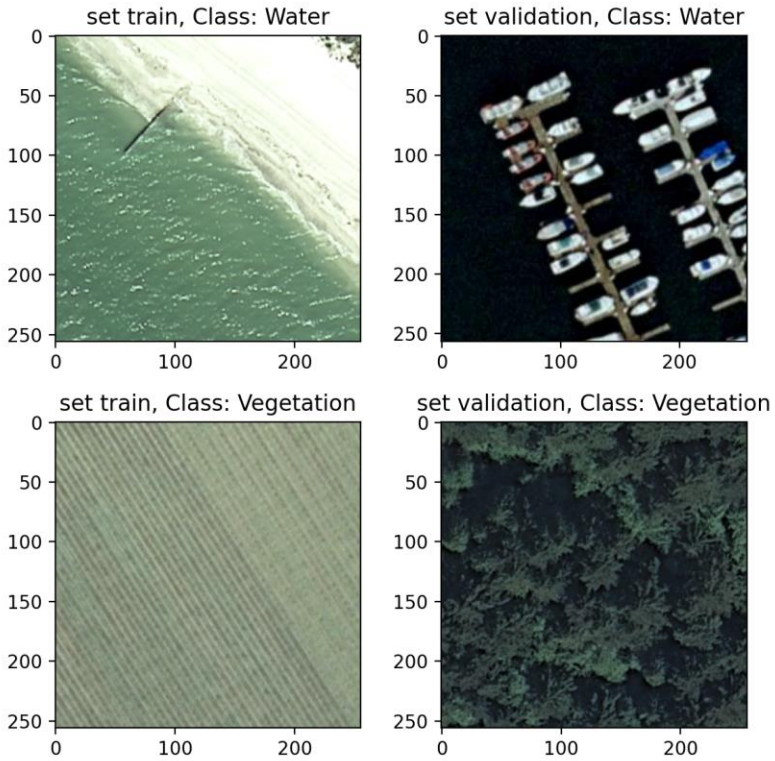
Remote sensing is the observation of a target by a device separated from it by some distance.(Barret and Curtis, 1976, p.3)

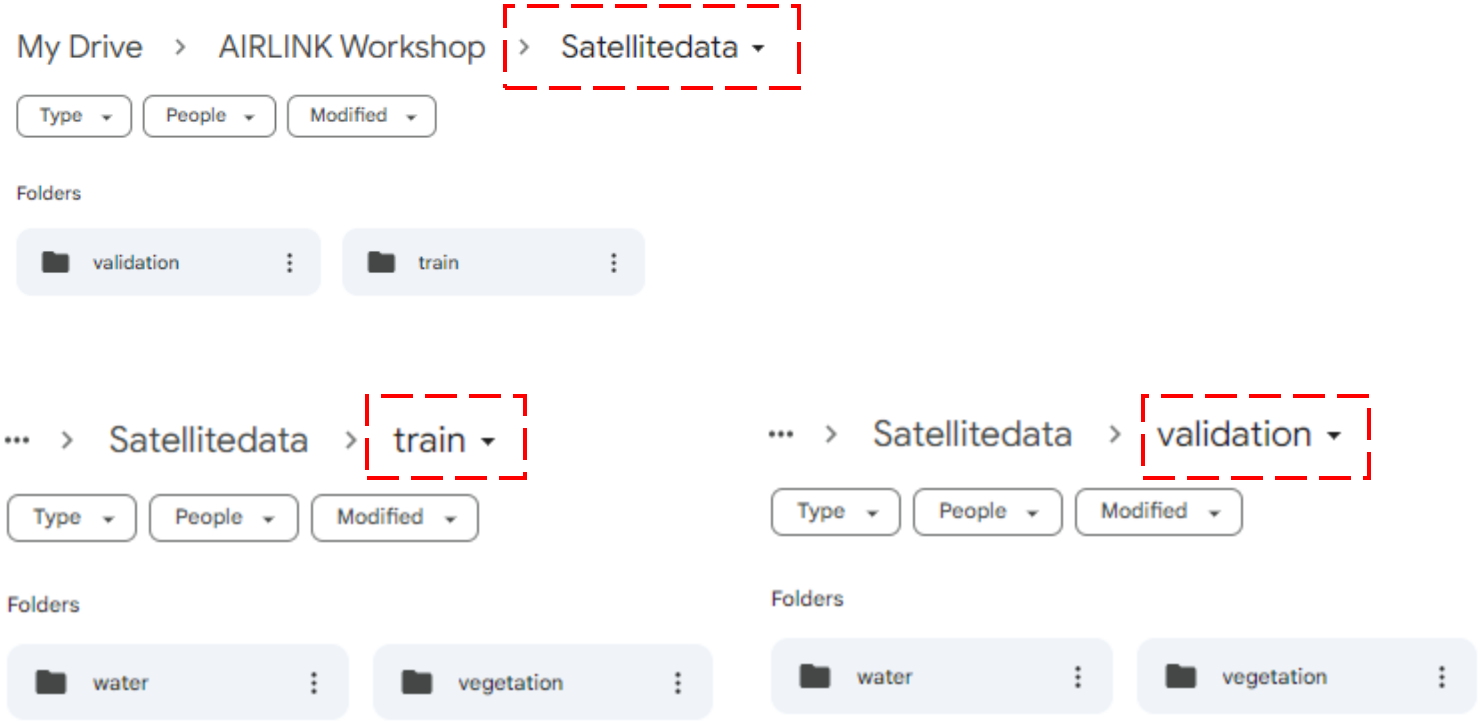


(CRISP, 2023)
(Abeer, 2022)
(Vali et al. 2020)

Datasets

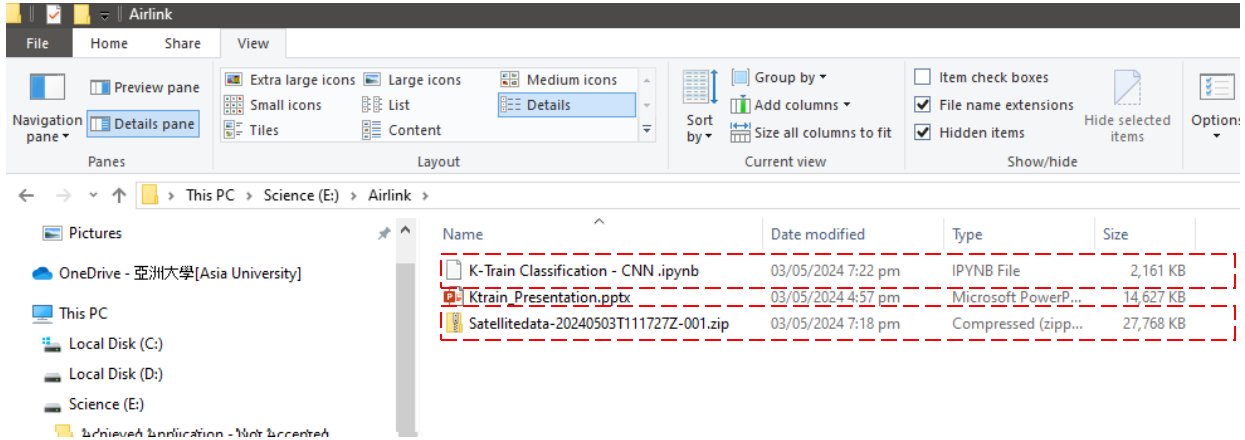
Satellite Images: Water | Vegetation



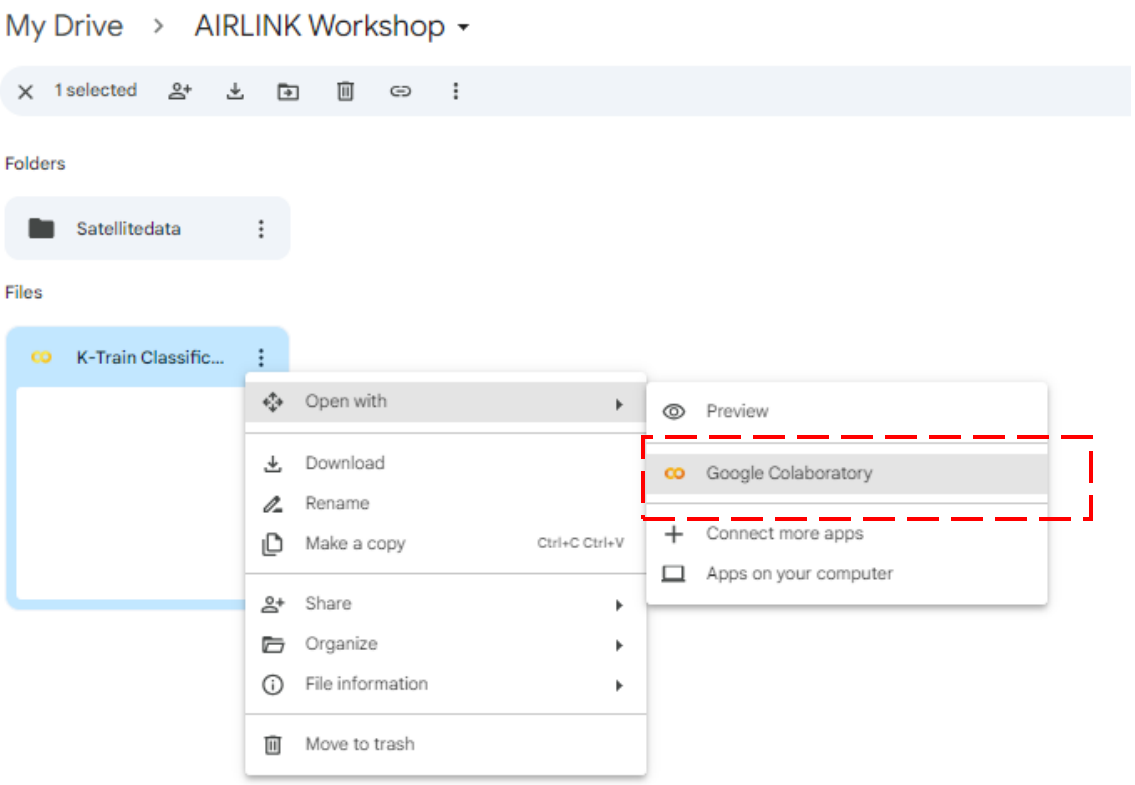


Datasets

Upload the code and datasets to GDrive

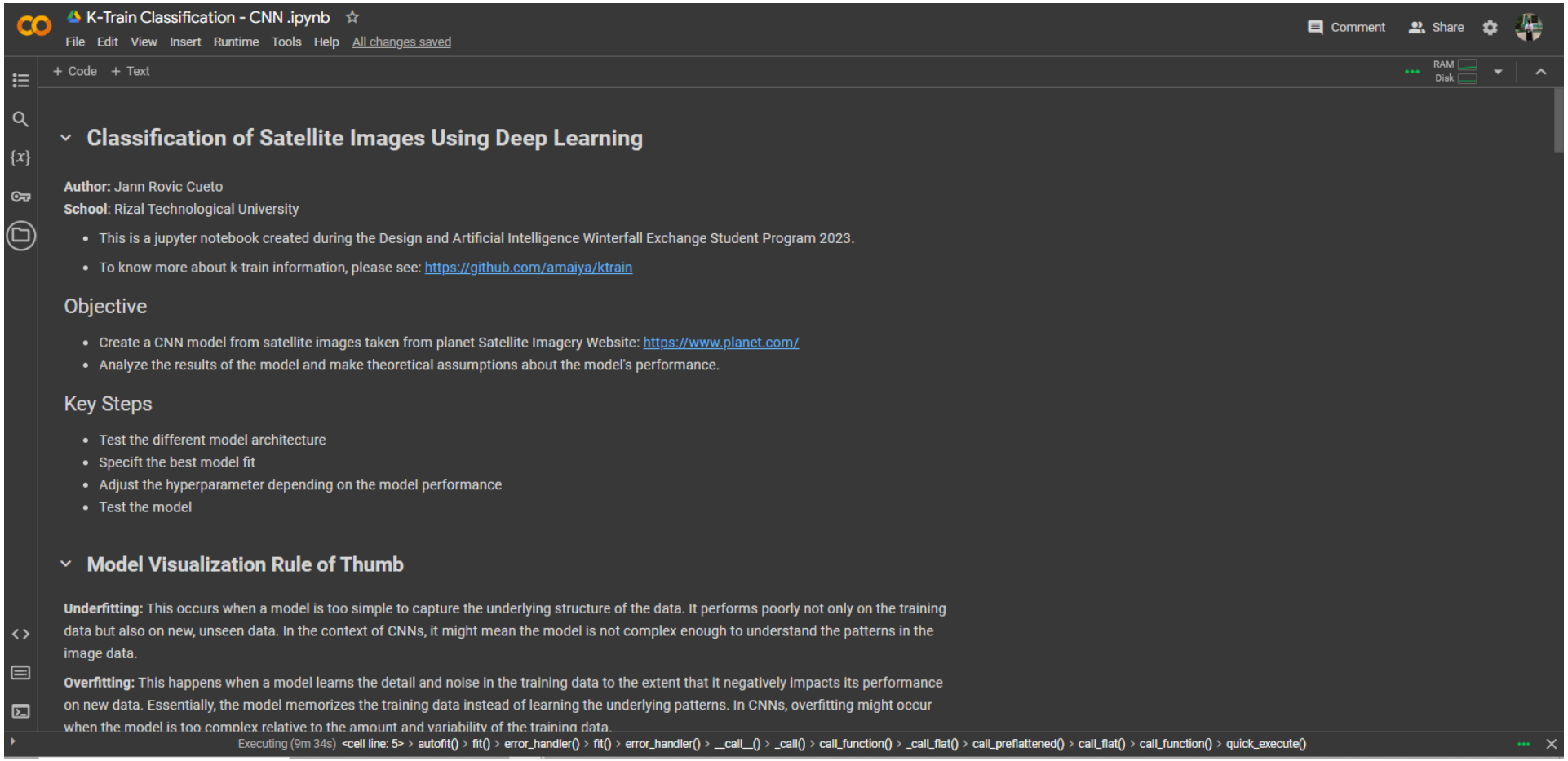


Open the Google Colab



Jupyter Notebook

Interface



Jupyter Notebook

Epoch and Batch Size

K-Train Classification - CNN .ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment Share

+ Code + Text

III. Hyperparameter Setting

```
[17] # Batch size can be adjusted. Depends on the dataset. It also depends on the GPU memory of the computer also.
     # In google colab the alloted maximum batch_size if you not subscribed is 16.
     learner = ktrain.get_learner(model=model, train_data=train_data,val_data=test_data,
                                  workers=8, use_multiprocessing=False, batch_size=8) # It can be 8, 16,32
```

```
[ ] # Freezes the first 15 images this helps to minimize the overfitting if the model undergoes it. Use this only as needed.
    # You can adjust the number of freeze you want.

    learner.freeze(freeze_range=15)
```

Epoch refers to one complete pass through the entire training dataset. During each epoch, the model is trained on the entire dataset in batches

- Think of an epoch as a full cycle of your model's learning process. It's like going through all your study materials once before an exam. During each epoch, your model studies the entire training dataset, learns from it, adjusts its understanding, and improves its performance. Just as you might need to review your notes multiple times to fully understand a topic, your model might need to go through several epochs to grasp the patterns in the data and perform well on new tasks

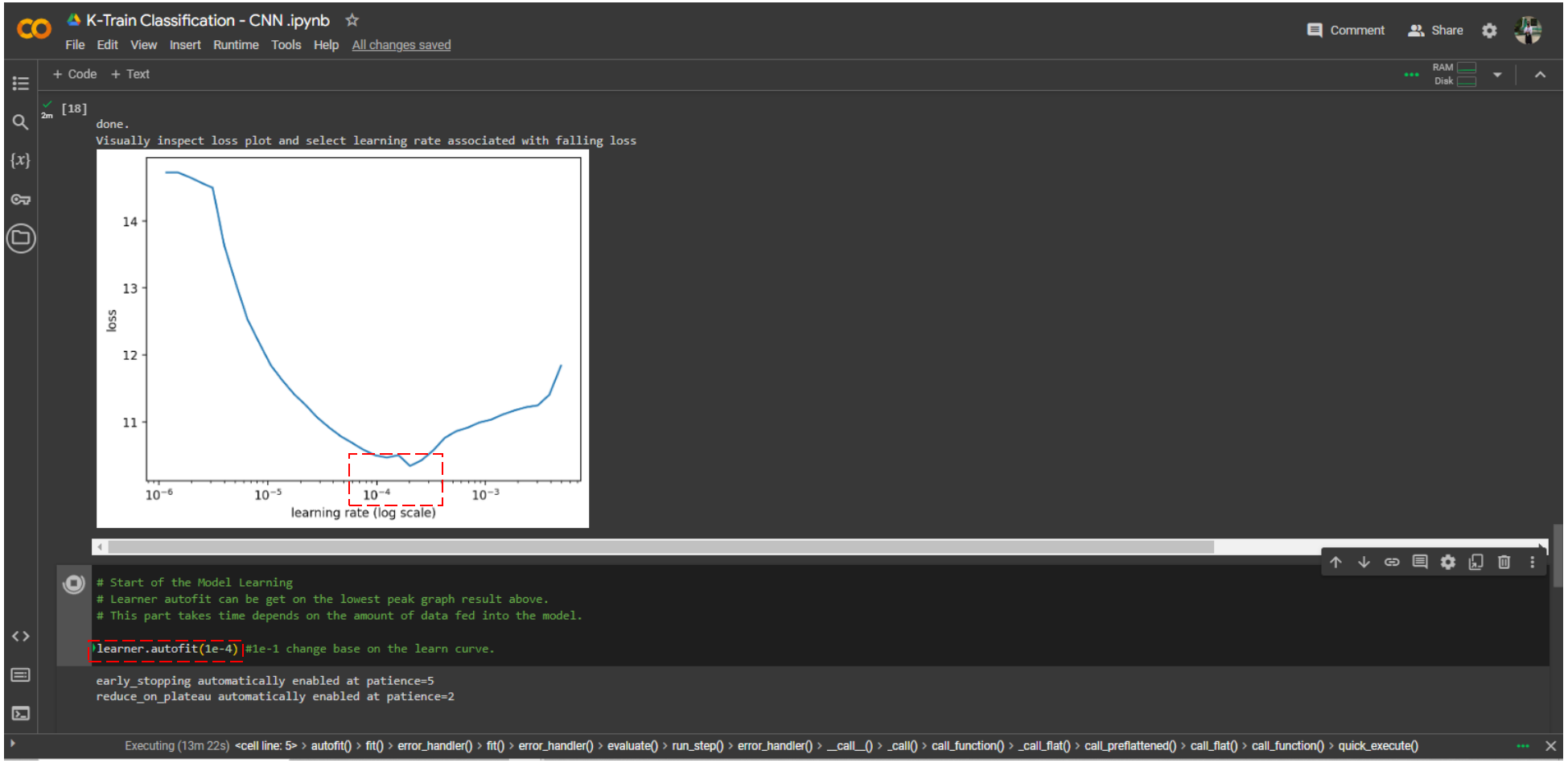
```
[18] # Epoch
     learner.lr_find(show_plot=True, max_epochs=5)# also this part can be adjusted.
```

```
simulating training for different learning rates... this may take a few moments...
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3103: UserWarning: You are saving your model as an HDF5 file via `model.save()`. This file format is considered legacy. We recommend using instead
  saving_api.save_model(
Epoch 1/5
15/15 [==============================] - 33s 2s/step - loss: 13.5753 - accuracy: 0.5083
Epoch 2/5
15/15 [==============================] - 31s 2s/step - loss: 9.7221 - accuracy: 0.5000
Epoch 3/5
15/15 [==============================] - 34s 2s/step - loss: 24.3235 - accuracy: 0.6083
Epoch 4/5
15/15 [==============================] - 10s 470ms/step - loss: 306.8983 - accuracy: 0.4250
```

Executing (11m 23s) <cell line: 5> > autofit() > fit() > error_handler() > fit() > error_handler() > evaluate() > run_step() > error_handler() > __call__() > _call() > call_function() > _call_flat() > call_preflattened() > call_flat() > call_function() > quick_execute()

Jupyter Notebook

Change the learning fit base on the result

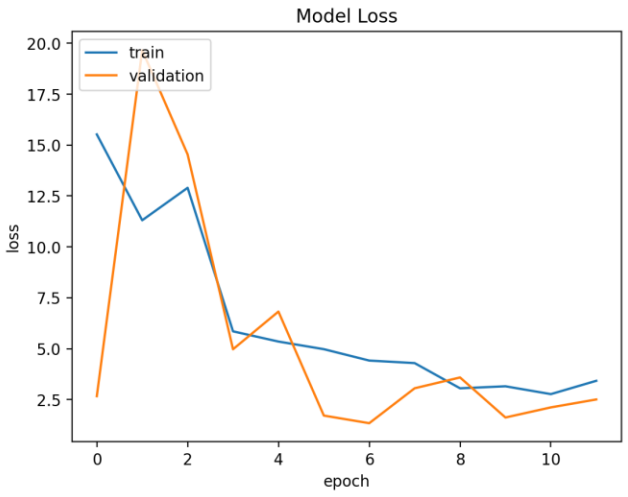


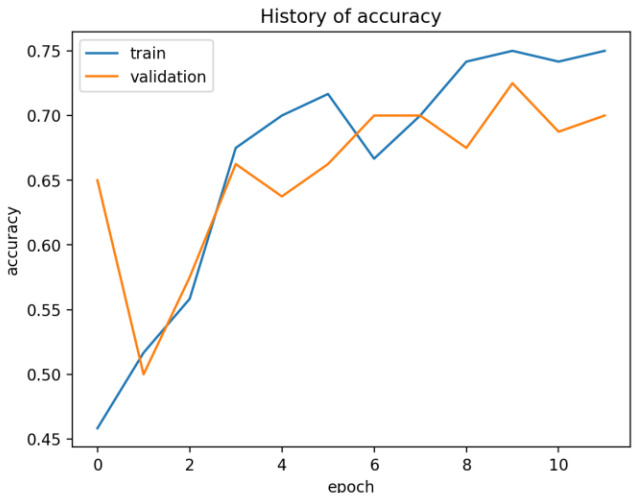
Jupyter Notebook

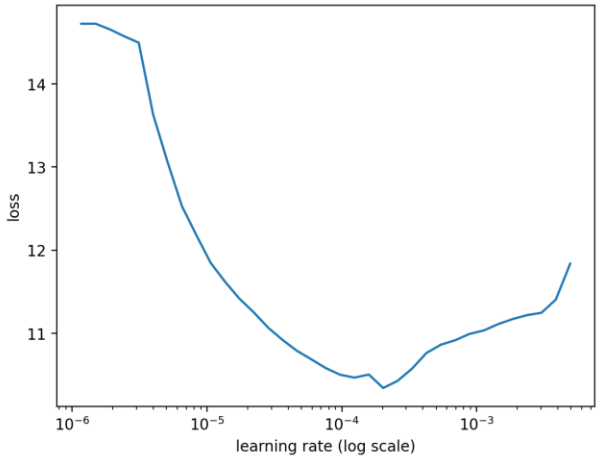
ResNet-50

```
[21] # Plot the Metric Performance
     learner.validate(class_names=preproc.get_classes())

     3/3 [==============================] - 24s 6s/step
                   precision    recall  f1-score   support

       vegetation       0.74      0.62      0.68        40
            water       0.67      0.78      0.72        40

         accuracy                           0.70        80
        macro avg       0.70      0.70      0.70        80
     weighted avg       0.70      0.70      0.70        80

     array([[25, 15],
            [ 9, 31]])
```

```
# Print the accuracy and error/loss of the model
loss, acc = learner.model.evaluate_generator(learner.val_data,
                                             steps=len(learner.val_data))
print('final loss:%s, final score:%s' % (loss, acc))

<ipython-input-24-3c079323fdb7>:2: UserWarning: `Model.evaluate_generator` is de
  loss, acc = learner.model.evaluate_generator(learner.val_data,
final loss:1.3391746282577515, final score:0.699999988079071
```





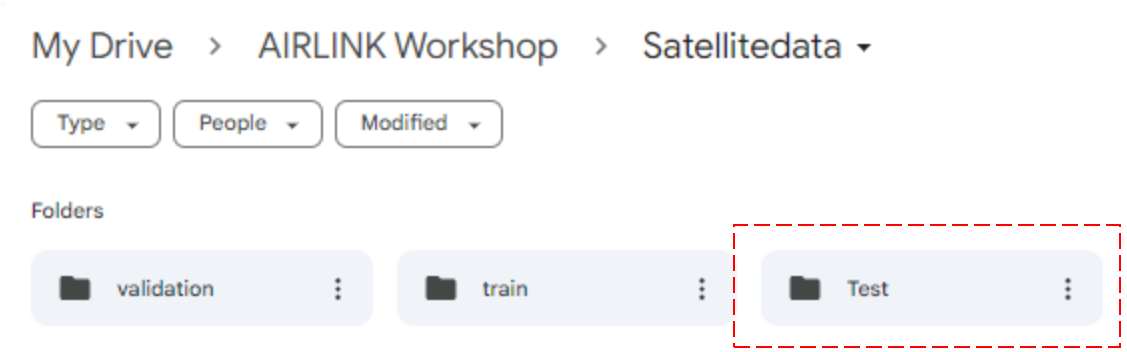


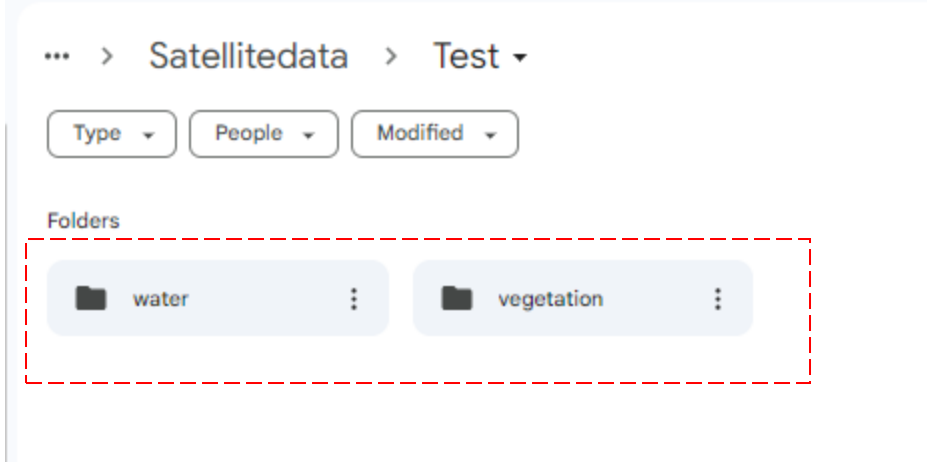
Final Score: 0.699999988079071
Final Loss: 1.3391746282577515

Jupyter Notebook

How to do the test?

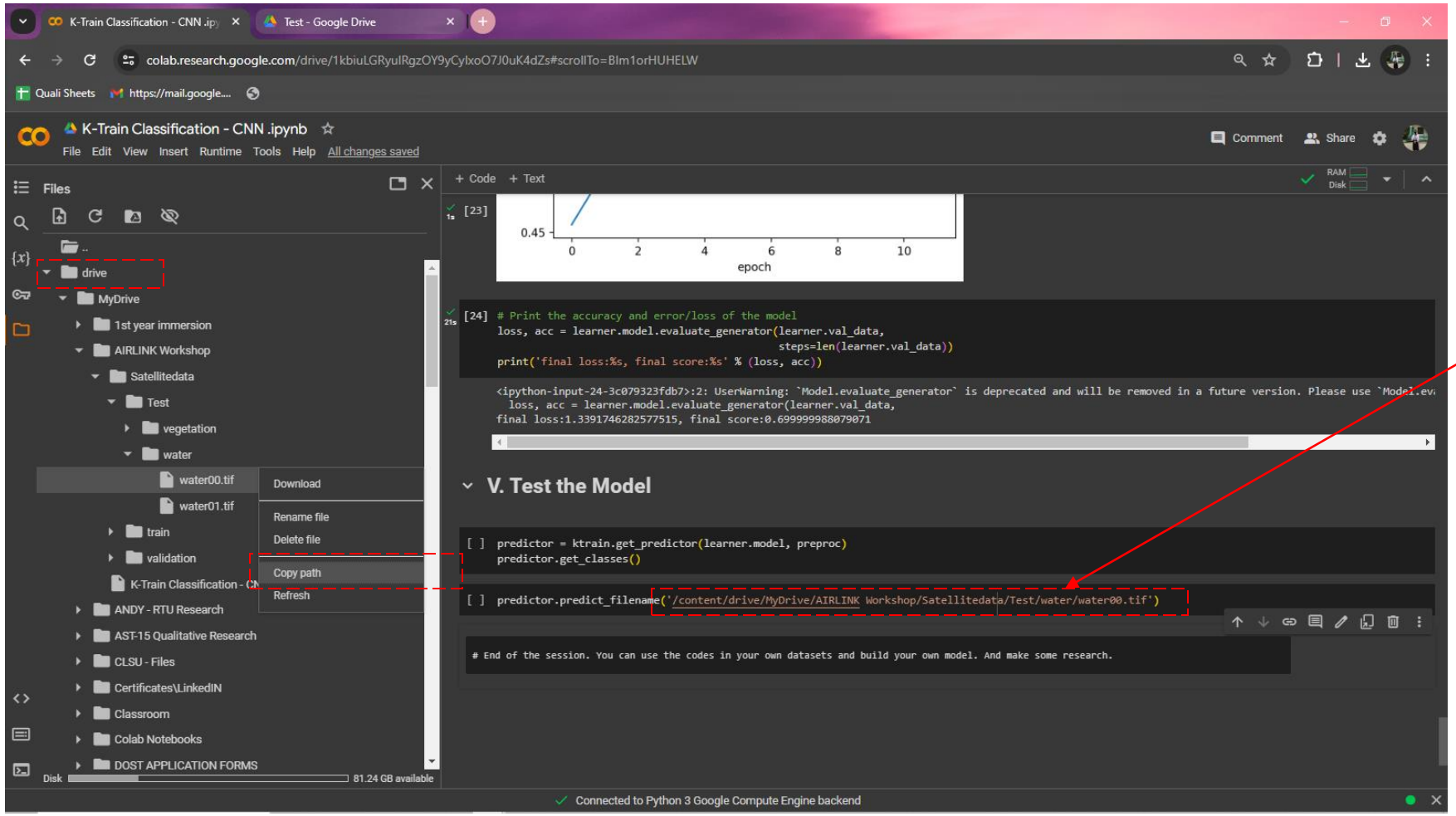
If you do not have test folder, you need to make new folder (same with train and validation, have water and vegetation folder inside)





Jupyter Notebook

How to do Test?



Remember to consider the directory

Thank you!

Artificial Intelligence is diverse, and so as researchers!

Contact Information:

https://www.facebook.com/jannrovic.cueto.3

https://github.com/roviccueto123

+63 967 432 2416

2022-104684@rtu.edu.ph

Jann Rovic Cueto

Astro_contis